

CLAIMS

I claim:

- 1 1. A method comprising:
 - 2 pruning local graphs representing local problems, the local problems
 - 3 corresponding to separately compilable components in a software program, each
 - 4 of the local graphs having edges and vertices, each edge having a transfer
 - 5 function, each vertex having a value, values of each of the local graph forming a
 - 6 lattice under a partial ordering.
- 1 2. The method of claim 1 wherein pruning the local graphs
 - 2 comprising:
 - 3 associating a use attribute to each one of the vertices in each of the local
 - 4 graphs, the use attribute being asserted for each vertex reachable from a named
 - 5 vertex;
 - 6 associating an affect attribute to each one of the vertices in each of the
 - 7 local graphs, the affect attribute is asserted for a vertex if a named vertex is
 - 8 reachable from the former vertex; and
 - 9 pre-solving a subgraph of each of the local graphs, the subgraph including
 - 10 subgraph edges, each of the subgraph edges connecting a tail vertex to a head
 - 11 vertex, the tail vertex having a negated use attribute.

1 3. The method of claim 2 wherein pruning the local graphs further
2 comprising:
3 shrinking the local graphs.

1 4. The method of claim 3 further comprising solving a global problem
2 to optimize a recompilation of the separately compilation components by an inter-
3 procedural analysis (IPA) solver, the global problem being represented by a global
4 graph formed from the pruned local graphs.

5 5. The method of claim 4 wherein pruning the local graphs further
6 comprising:
7 determining final edges and vertex values of the local graphs to be sent to
8 IPA solver; and
9 sending the final edges and vertex values to the IPA solver, the final edges
10 and vertex values forming the pruned local graphs.

1 6. The method of claim 2 wherein associating the use attribute
2 comprises:
3 negating use attributes for all vertices in the local graph; and
4 invoking a mark use operation on u for each named vertex u in the local
5 graph.

1 7. The method of claim 6 wherein invoking the mark use operation on
2 u comprises:

3 asserting the use attribute associated with u if the use attribute is negated;
4 and

5 recursively invoking the mark use operation on v for each edge connecting
6 the named vertex u to a vertex v.

1 8. The method of claim 2 wherein associating the affect attribute
2 comprises:

3 negating use attributes for all vertices in the local graph;

4 invoking a mark affect operation on y for each named vertex y in the local
5 graph.

1 9. The method of claim 8 wherein invoking the mark affect operation
2 on y comprises:

3 asserting the use attribute associated with y if the use attribute is negated;
4 and

5 recursively invoking the mark affect operation on x for each edge
6 connecting the vertex x to a named vertex y.

1 10. The method of claim 2 wherein pre-solving the subgraph
2 comprises:

3 finding a greatest fix-point solution to the subgraph.

1 11. The method of claim 3 wherein shrinking comprises:

2 removing an incoming edge having a head value of a lattice-bottom.

1 12. The method of claim 3 wherein shrinking further comprises:

2 transforming a subgraph having first and second edges, the first and
3 second edges having first and second functions, the first edge connecting a first
4 vertex to an anonymous vertex having a value v , the second edge connecting the
5 anonymous vertex to a second vertex having a value w .

1 13. The method of claim 12 wherein transforming comprises:

2 removing the anonymous vertex;

3 removing first and second edges;

4 adding a third edge having a third function and connecting the first and
5 second vertices, the third function being combined by the first and second
6 functions; and

7 changing value of the second vertex to a lattice meet of the second
8 function of the value v and the value w.

1 14. The method of claim 5 wherein determining the final edges and
2 vertex values comprises:

3 determining each of the final edges as edge having asserted use and affect
4 attributes for tail and head vertices, respectively; and

5 eliding each of the vertex values having a top value.

1 15. A computer program product comprising:

2 a machine useable medium having computer program code embedded
3 therein, the computer program product having:

4 computer readable program code to prune local graphs representing local
5 problems, the local problems corresponding to separately compilable components
6 in a software program, each of the local graphs having edges and vertices, each
7 edge having a transfer function, each vertex having a value, values of each of the
8 local graph forming a lattice under a partial ordering.

1 16. The computer program product of claim 15 wherein the computer
2 readable program code to prune the local graphs comprising:

3 computer readable program code to associate a use attribute to each one of
4 the vertices in each of the local graphs, the use attribute being asserted if there is
5 an edge connecting a named vertex to the each one of the vertices;

6 computer readable program code to associate an affect attribute to each
7 one of the vertices in each of the local graphs, the affect attribute is asserted if
8 there is an edge connecting the each one of the vertices to a named vertex; and

9 computer readable program code to pre-solve a subgraph of each of the
10 local graphs, the subgraph including subgraph edges, each of the subgraph edges
11 connecting a tail vertex to a head vertex, the tail vertex having a negated use
12 attribute.

1 17. The computer program product of claim 16 wherein the computer
2 readable program code to prune the local graphs further comprising:

3 computer readable program code to shrink the local graphs.

1 18. The computer program product of claim 15 further comprising:

2 computer readable program code to solve a global problem to optimize a
3 recompilation of the separately compilation components by an inter-procedural
4 analysis (IPA) solver, the global problem being represented by a global graph
5 formed from the pruned local graphs.

1 19. The computer program product of claim 18 wherein the computer
2 readable program code to prune the local graphs further comprising:

3 computer readable program code to determine final edges and vertex
4 values of the local graphs to be sent to IPA solver; and
5 computer readable program code to send the final edges and vertex values
6 to the IPA solver, the final edges and vertex values forming the pruned local
7 graphs.

1 20. The computer program product of claim 16 wherein the computer
2 readable program code to associate the use attribute comprises:

3 computer readable program code to negate use attributes for all vertices in
4 the local graph;

5 computer readable program code to invoke a mark use operation on u for
6 each named vertex u in the local graph.

1 21. The computer program product of claim 19 wherein the computer
2 readable program code to invoke the mark use operation on u comprises:

3 computer readable program code to assert the use attribute associated with
4 u if the use attribute is negated; and

5 computer readable program code to recursively invoke the mark use
6 operation on v for each edge connecting the named vertex u to a vertex v.

1 22. The computer program product of claim 16 wherein the computer
2 readable program code to associate the affect attribute comprises:

3 computer readable program code to negate use attributes for all vertices in
4 the local graph; and

5 computer readable program code to invoke a mark affect operation on y
6 for each named vertex y in the local graph.

1 23. The computer program product of claim 22 wherein the computer
2 readable program code to invoke the mark affect operation on y comprises:

3 computer readable program code to assert the use attribute associated with
4 y if the use attribute is negated; and

5 computer readable program code to recursively invoke the mark affect
6 operation on x for each edge connecting the vertex x to a named vertex y.

1 24. The computer program product of claim 16 wherein the computer
2 readable program code to pre-solve the subgraph comprises:

3 computer readable program code to find a greatest fix-point solution to the
4 subgraph.

1 25. The computer program product of claim 17 wherein the computer
2 readable program code to shrink comprises:

3 computer readable program code to remove an incoming edge having a
4 head value of a lattice-bottom.

1 26. The computer program product of claim 17 wherein the computer
2 readable program code to shrink further comprises:

3 computer readable program code to transform a subgraph having first and
4 second edges, the first and second edges having first and second functions, the
5 first edge connecting a first vertex to an anonymous vertex having a value v , the
6 second edge connecting the anonymous vertex to a second vertex having a value
7 w .

1 27. The computer program product of claim 26 wherein the computer
2 readable program code to transform comprises:

3 computer readable program code to remove the anonymous vertex;

4 computer readable program code to remove first and second edges;

5 computer readable program code to add a third edge having a third
6 function and connecting the first and second vertices, the third function being
7 combined by the first and second functions; and

8 computer readable program code to change value of the second vertex to a
9 lattice meet of the second function of the value v and the value w .

1 28. The computer program product of claim 19 wherein the computer
2 readable program code to determine the final edges and vertex values comprises:

3 computer readable program code to determine each of the final edges as
4 edge having asserted use and affect attributes for tail and head vertices,
5 respectively; and

6 computer readable program code to elide each of the vertex values having
7 a top value.

1 29. A system comprising:
2 a processor; and
3 a memory coupled to the processor to store instruction code, the
4 instruction code, when executed by the processor, causing the processor to:
5 prune local graphs representing local problems, the local problems
6 corresponding to separately compilable components in a software
7 program, each of the local graphs having edges and vertices, each edge
8 having a transfer function, each vertex having a value, values of each of
9 the local graph forming a lattice under a partial ordering.

1 30. The system of claim 29 wherein the instruction code causing the
2 processor to prune the local graphs causes the processor to:
3 associate a use attribute to each one of the vertices in each of the local
4 graphs, the use attribute being asserted if there is an edge connecting a named
5 vertex to the each one of the vertices;

6 associate an affect attribute to each one of the vertices in each of the local
7 graphs, the affect attribute is asserted if there is an edge connecting the each one
8 of the vertices to a named vertex; and

9 pre-solve a subgraph of each of the local graphs, the subgraph including
10 subgraph edges, each of the subgraph edges connecting a tail vertex to a head
11 vertex, the tail vertex having a negated use attribute.

1 31. The system of claim 30 wherein the instruction code causing the
2 processor to prune the local graphs further causes the processor to:

3 shrink the local graphs.

1 32. The system of claim 31 wherein the instruction code further
2 causing the processor to:

3 solve a global problem to optimize a recompilation of the separately
4 compilation components by an inter-procedural analysis (IPA) solver, the global
5 problem being represented by a global graph formed from the pruned local
6 graphs.

7 33. The system of claim 32 wherein the instruction code causing the
8 processor to prune the local graphs further causes the processor to:

9 determine final edges and vertex values of the local graphs to be sent to
10 IPA solver; and

11 send the final edges and vertex values to the IPA solver, the final edges
12 and vertex values forming the pruned local graphs.

1 34. The system of claim 30 wherein the instruction code causing the
2 processor to pre-solve the subgraph causes the processor to:

3 find a greatest fix-point solution to the subgraph.

1 35. The system of claim 31 wherein the instruction code causing the
2 processor to shrink causes the processor to:

3 remove an incoming edge having a head value of a lattice-bottom.

1 36. The system of claim 35 wherein the instruction code causing the
2 processor to shrink further causes the processor to:

3 transform a subgraph having first and second edges, the first and second
4 edges having first and second functions, the first edge connecting a first vertex to
5 an anonymous vertex having a value v , the second edge connecting the
6 anonymous vertex to a second vertex having a value w .

1 37. The system of claim 36 wherein the instruction code causing the
2 processor to transform causing the processor to:

3 remove the anonymous vertex;

4 remove first and second edges;

5 add a third edge having a third function and connecting the first and
6 second vertices, the third function being combined by the first and second
7 functions; and

8 change value of the second vertex to a lattice meet of the second function
9 of the value v and the value w.

1 38. The system of claim 33 wherein the instruction code causing the
2 processor to determine the final edges and vertex values causes the processor to:

3 determine each of the final edges as edge having asserted use and affect
4 attributes for tail and head vertices, respectively; and

5 elide each of the vertex values having a top value.